## **Data Types**

**The goal for this exercise** is to understand what simple data types are available in C#, so that you can choose the best type for your variables.

**What you need to do for this exercise:** If it hasn’t been created already, then you should create a small C# class named **Data\_Type\_Explanations**, and inside it, add a method named **RunExercise**, in the starter project that has been provided to you. In that method, you need to:

1. Create each of the following data types (as a local variable), store a reasonable value into that variable, and then print out the contents of that variable.
	1. Include a comment about what the minimum and maximum value of each type is
		1. The **string** type doesn’t have minimum / maximum values. Instead create a **string** variable that holds a string (any string you want – pick one), and then prints it out.
	2. Make sure to print out all the numbers with something after the decimal point (float, double, decimal) with EXACTLY two decimal places.
2. For the **string** variable you should print out:
	1. the string itself
	2. an all-uppercase version of that string using the **ToUpper()** method. This method is documented on the MSDN website (at [http://msdn.microsoft.com/en-us/library/ewdd6aed(v=vs.110).aspx](http://msdn.microsoft.com/en-us/library/ewdd6aed%28v%3Dvs.110%29.aspx)), and that documentation includes an example of using the method (warning: the example is buried in some other code that you don’t need to understand, so you may need to dig for it a bit)
	3. an all-lowercase version of the string using the **ToLower()** method. It is left as an exercise for you to find whatever documentation you need to successfully utilize this method.
3. After doing the above items, include a comment the prefix **u** means. In other words, what does the ‘u’ mean in ‘ushort’, ‘uint’, and ‘ulong’? Make sure to both explain what the ‘u’ is short for, and to clearly explain what sort of values you can expect to store in a data type with the ‘u’ prefix
4. You also need to include a comment as to the general differences between float/double and decimal
	1. You need to specifically list at least one sort of situation wherein you'd want to use decimal, rather than a float or double.
5. You also need to include (in a comment, at the bottom of the RunExercise method), answers to each of the questions listed at the bottom of this document.

You may need to consult your textbook (see the Lessons page on the class website for pages/sections), and/or the C# language specification (included in Visual Studio, and available online), and/or the internet to find these answers.

* sbyte
* byte
* short
* ushort
* **int**
* uint
* long
* ulong
* char
* float
* **double**
* decimal
* **bool** // You haven't seen this yet, but it should be familiar// from other languages – it's true/false
* **string**

Example:

public void RunExercise()

{

 sbyte x = 2; // Can store from -128 to +127

 Console.WriteLine( "Value of x is: " + x);

// Next, use the byte, etc.

// The "u" prefix means...

// Float/double vs. decimal:...

}

When you print out the numbers with something after the decimal point (float, double, decimal, you need to print out EXACTLY two decimal places. You can do this by specifying which thing to print out (using the normal {**x**} syntax, where **x** is the index of the thing to print), followed by a colon, followed by a sample format, like so:

float f = 1.117f;

double d = 1.217;

decimal de = 1.314m;

Console.WriteLine("f: {0:0.00} d:{1:0.00} de:{2:0.00}",
 f, d, de);

The above code snippet will print out (note that it rounds the number – 1.117 is printed as 1.1**2**, while 1.314 is rounded down when printing, to 1.3**1**)

**f: 1.12 d:1.22 de:1.31**

**You are required to print out all floats, doubles, and decimals to exactly two decimal points throughout this course, unless otherwise stated.**

Additional Questions to answer:

 In a comment at the bottom of your RunExercise method, you should answer each of the following questions. Make sure to select appropriate data type(s) based on efficiency of data storage (i.e., not using more space than you need to), and ease of maintenance (i.e., will it be easier to write a program that will be correct, and easy to change/fix over time), from the list of data types above.

 For each answer, make sure to include a quick (1-2 sentence) explanation about WHY you’ve chosen the data type(s) that you’ve chosen. This part (thinking about which data type to use, and being able to explain/justify it) is probably the most important part of this exercise.

 Make sure to include the question number in the comment, so it’s clear which question you’re answering.

1. What variable type(s) would you use to store the number of students enrolled in a class at Cascadia Community College? Keep in mind that classes are limited to 30 students (48 for a ‘learning community’ class).
2. What variable type(s) would you use to store a student’s current tuition balance? Keep in mind that a student may not have paid their tuition yet (in which case the balance is a positive number), or they may have overpaid (in which case the balance will be a negative number), and that the balance is always in US dollars.
3. What variable type(s) would you use to store the result of a scientific computation? Keep in mind that having extremely precise results is very important, and that occasionally having to deal with round-off errors is ok, as long as the data has the maximum number of digits of precision in other cases.
4. What variable type(s) would you use to store a Cascadia Student ID (SID)? If you need an example of a SID, you can check the back of your Cascadia student ID card – it’s the number that starts with “9600…”, and is 9 digits long.
5. Let’s say you’ve written a program that will send email to all of the students in a given class at Cascadia. What variable type(s) would you use to store, in memory, one of two values: one value will mean that your program’s attempt at emailing all of the students in a class succeeded, the other value will mean that at least one of the emails was not successfully sent.
6. Let’s say you’ve written a program that will send email to all of the students in a given class at Cascadia. Let’s say that you want to reuse this program for multiple classes, so you decide to store the full name of the class in a variable. What variable type(s) would you use to store this information?
	1. Some examples of class names:
		1. BIT 142
		2. ENG 101
		3. MATH 141
		4. etc